모의해킹 결과 보고서

작 성 자: 이 길 형

목차

1. 개요

1.1 모의해킹 정의

본 모의해킹 진단은 워게임용으로 제작된 웹 애플리케이션을 대상으로 취약점을 도출 · 분석하고, 이에 대한 대응 방안을 수립하기 위해 수행되었습니다. 실제 해커와 유사한 환경 및 조건, 그리고 해킹 기법을 바탕으로 모의 침투를 수행하였으며, 이를 통해 발견된 취약점에 대해 사전 예방 조치를 마련하는 것을 주요 목표로 합니다.

1.2 수행 기간 및 인원

수행 기간

단계	구분	일정
모의해킹	사전환경조사	2025.05.29~2025.06.02
	내부 시스템 침투 테스트	2025.06.02~2025.06.11
	모의해킹 진단 보고서 작성	2025.06.11~2025.06.13

[표 1-1] 모의해킹 수행 기간

수행 인원

소속	성명	역할
코리아 IT 아카데미 Sunglass 조	이길형	취약점 진단 및 웹 해킹
	이재호	취약점 진단 및 웹 해킹
	유승민	로그 탐지

[표 1-2] 모의해킹 수행 인원

1.3 수행 대상 및 장소

수행 대상

구분	대상 IP	서비스
모의해킹	192.168.5.160	워게임

[표 1-3] 모의해킹 수행 대상

수행 장소

구분	수행자 네트워크 대역	장소
모의해킹	192.168.0.0/16	코리아 IT 아카데미 강의실

[표 1-4] 모의해킹 수행 장소

1.4 수행 단계별 방법

[그림 1-1] 수행 단계

수행 단계	설명
정보 수집	대상에 대한 서버/네트워크/서비스에 대한 불필요한 서비스 접근 가능성, 외부에서 파악할 수 있는 정보들을 수집하는 단계
취약점 수집	적합한 취약점 스캔 도구를 이용하여 발생할 수 있는 취약점에 대한 정보를 수집하는 단계
침투 단계	취약점 수집 단계를 통해 획득한 정보를 기반으로 수동점검을 통해 내부 시스템까지 침투할 가능성이 있는지 시나리오 기반으로 접근하는 단계
상세 분석	취약점이 도출되었을 경우에 공격에 의해서 보안 위협이 시스템 및 비즈니스 측면에서 어느 정도의 영향을 줄 수 있는지 분석하는 단계
보고서 작성	도출된 취약점에 대한 총평/영향도/상세분석/보안가이드가 포함된 보고서를 작성하는 단계

[표 1-5] 수행 단계별 방법

1.5 침투 시나리오

[그림 1-2] 침투 시나리오

① 네트워크 대역의 정보를 획득합니다.

② 웹 서버 공격 시도합니다.

③ WEB 서버/애플리케이션에서 발생할 수 있는 취약점을 이용하여 내부 서버 침투를 시도합니다.

④ 데이터베이스/로그서버 등에 침투를 하여 개인정보/사내 주요 정보를 획득합니다.

접근법은 모두 동일하나 방화벽 내부 네트워크 대역에서 진행됩니다.

1.6 점검 항목

점검항목은 KISA 의 주요정보통신기반시설 기술적 취약점 분석 · 평가 방법 상세가이드를 기반으로 구성하였습니다.

순번	코드	분류	위험도	점검 내용
1	BO	버퍼 오버플로우	상	사용자가 입력한 파라미터 값의 문자열 길이 제한 확인
2	SI	SQL 인젝션	상	웹페이지 내 SQL 인젝션 취약점 존재 여부 점검
3	DI	디렉터리 인덱싱	상	특정 디렉터리에 초기 페이지의 파일이 존재하지 않을 때 자동으로 디렉터리 리스트를 출력하는지 여부
4	IL	정보 누출	상	웹 서비스 시 불필요한 정보가 노출되는지 여부 점검
5	XS	크로스사이트 스크립팅	상	웹 사이트 내 크로스사이트 스크립팅 취약점 존재 여부 점검

6	SF	세션 고정	상	사용자 로그인 시 항상 일정하게 고정된 세션 ID 값을 발행하는지 여부 확인
7	FU	파일 업로드	상	웹 사이트의 게시판, 자료실 등에 조작된 Server Side Script 파일 업로드 및 실행 가능 여부 점검

[표 1-6] 점검 항목

1.7 점검 도구

구분	진단 도구	용도
모의해킹 취약점 점검 도구	OWASP ZAP	자동화된 웹 취약점 스캔
	Nmap	IP 와 포트, 운영체제, 서비스 정보 등을 스캔

[표 1-7] 점검 도구

2. 결과 요약

2.1 위험도 평가 기준

평가등급	기준
상 (High)	• 취약점을 통해 권한 상승 및 직접적인 관리자 권한 획득이 가능한 취약점 • 취약점을 통해 데이터 변조 및 서비스 가용성에 큰 영향을 줄 수 있는 취약점
중 (Medium)	• 취약점이 존재하나 권한 상승 및 관리자 권리자 권한 획득 어려운 취약점 • 추후 공격을 통해 권한 상승 및 서비스 가용성에 큰 영향을 줄 수 있는 잠재적인 취약점
하 (Low)	• 해당 취약점으로 인해 시스템에 영향을 주지는 않으나, 시스템에 대한 일부 정보를 수집할 수 있는 취약점

[표 2-1] 위험도 평가 기준

2.2 총평

전체 5 개 취약점이 발견되었으며 위험도 상(H): 3 개, 중(M): 0 개, 하(L): 0 개로 총 5 개의 취약점이 도출되었습니다. 주요 취약점은 아래와 같습니다.

[SQL 인젝션]

해당 워게임 웹에서 SQL Injection 취약점을 이용하여 사용된 테이블 내의 모든 정보를 열람할 수 있었으며, 나아가 해당 데이터베이스에 권한이 부여된 계정 정보 또한 확인할 수 있었습니다.

[크로스사이트 스크립팅]

해당 워게임 웹의 ./fileupload 경로에서 XSS 취약점 점검을 수행하였습니다. onMouseOver 이벤트를 통해 쿠키값을 탈취하여 계정을 변경하였습니다.

[파일 업로드]

해당 워게임 웹의 /.fileupload 경로에는 개발자가 지정한 파일의 확장자 이외의 다른 확장자의 파일을 업로드 하지 못하게 되어 있습니다. 하지만 확장자를 우회하게 되면 파일을 손쉽게 업로드 할 수 있는데 예를 들어 (.php) 파일을 (.php.kr) 파일로 변환하는 등 다양하게 업로드 가능합니다. 리버스 쉘을 통해 서버에 접근하여 데이터베이스에 접근이 가능한 계정 정보를 탈취하는데 성공하였습니다.

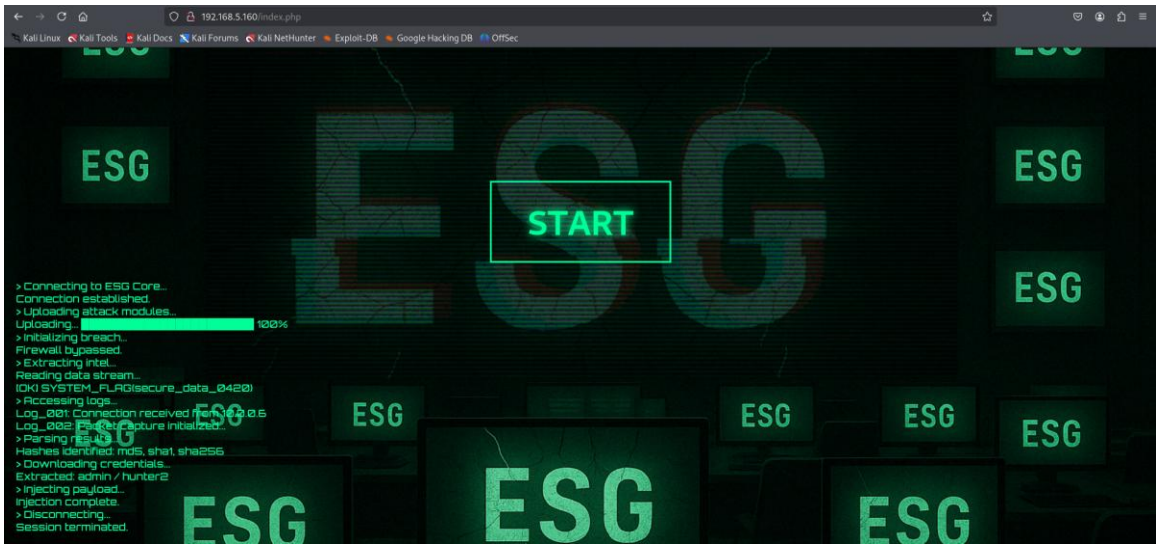
2.3 탐지된 주요 공격 요약

서비스	취약점	요약
워게임 문제	SQL 인젝션	SQL 쿼리문을 조작하여 공격자가 원하는 정보를 추출할 수 있다.
파일 업로드	크로스사이트 스크립팅	스크립트 구문을 출력할 수 있고 document.cookie 객체를 이용하여 사용자의 쿠키값을 탈취 가능하다.
	파일 업로드 취약점	업로드 페이지에 개발자가 지정한 파일 확장자 이외의 파일을 확장자를 우회하여 업로드 할 수 있고 리버스 쉘을 업로드하고 실행 가능하다.

[표 2-2] 탐지된 주요 공격 요약

3. 상세 설명

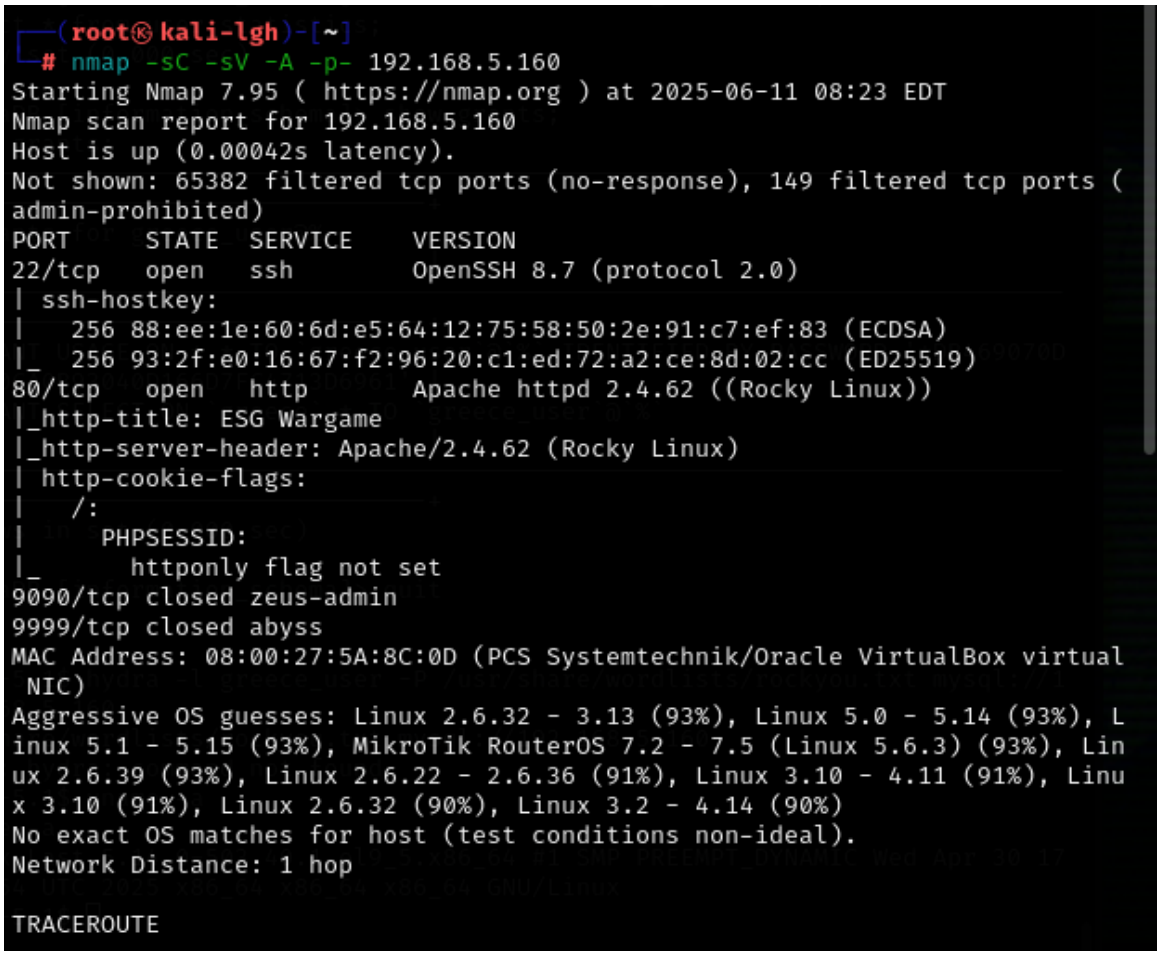
3.1 환경 분석



[그림 3-1] 워게임 웹

워게임 웹사이트의 각 페이지로 접근하게 되면 각 페이지들이 *.php 으로 저장되어 있는 것을 보이는데 이를 통해 본 서비스 페이지들이 php 로 구성되어 있음을 확인하였습니다.

다음으로 웹 서비스를 제공하는 서버 자체의 환경 구성에 대한 정보를 수집합니다.



[그림 3-2] nmap 을 이용한 Web Server 스캔

[그림 3-2]는 Network Scanning Tool 인 nmap 을 활용하여 워게임을 제공하는 서버에 스캔을 시도한 결과로 현재 서버는 Linux 기반이고 현재 2 개의 포트가 열려 있는 것을 확인하였습니다.

서비스 종류	상세 설명
Operating System	Linux
Port(open)	22/ssh
	80/tcp
MAC Address	08:00:27:5A:8C:0D

[표 3-1] 환경 분석 결과

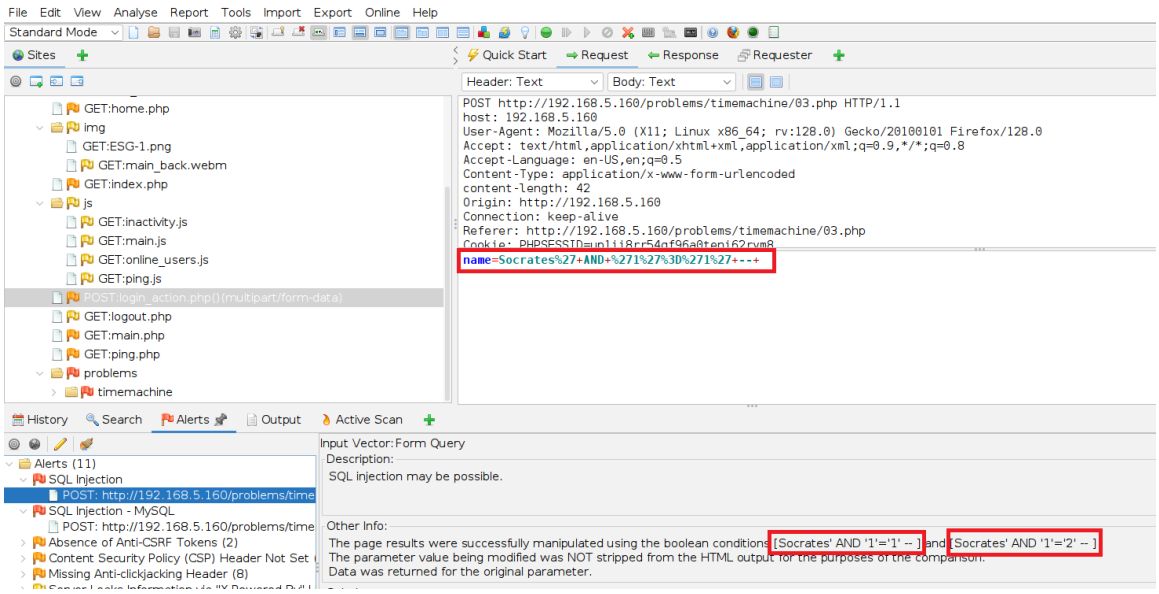
3.2 SQL 인젝션

3.2.1 취약점 개념 설명

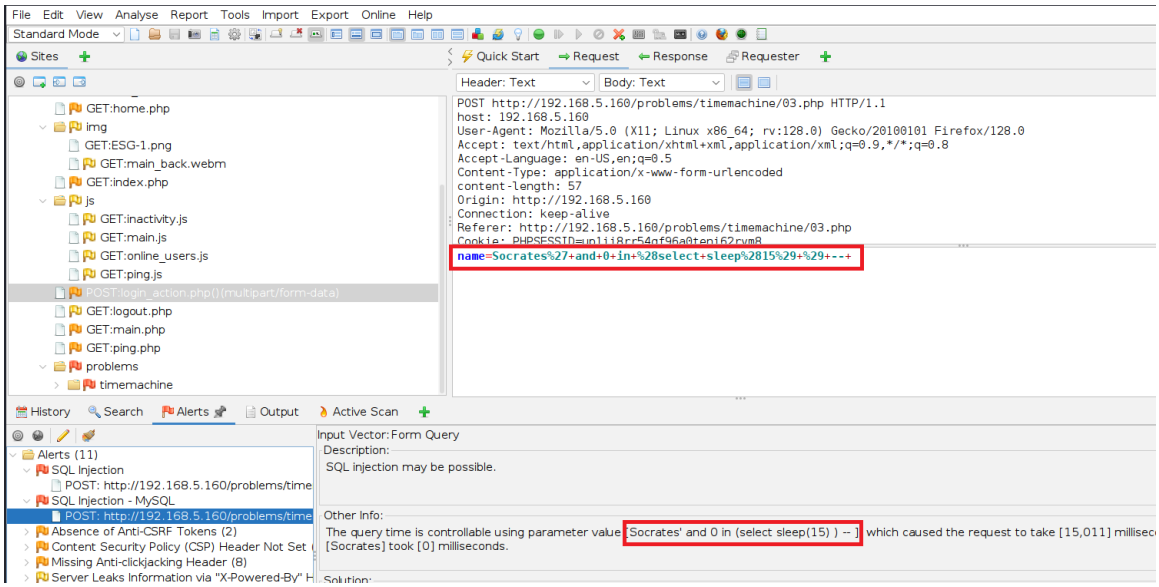
SQL(Structured Query Language)은 웹사이트나 프로그램이 데이터베이스에 접속해서 정보를 저장하거나 불러올 때 사용하는 언어입니다. SQL 은 인터프리터 언어의 일종이기 때문에, 사용자가 입력한 값이 프로그래머가 작성한 명령문과 함께 해석되어 실행됩니다. 그런데 이 입력값을 제대로 검사하지 않으면, 공격자가 일부러 이상한 값을 입력해 정상적인 동작을 방해할 수 있습니다. 이처럼 입력값에 조작된 명령을 숨겨서 데이터베이스가 원래와는 다르게 동작하게 만드는 공격을 SQL 인젝션(SQL Injection)이라고 합니다.

3.2.2 취약점 점검

(1) OWASP ZAP 을 이용한 점검



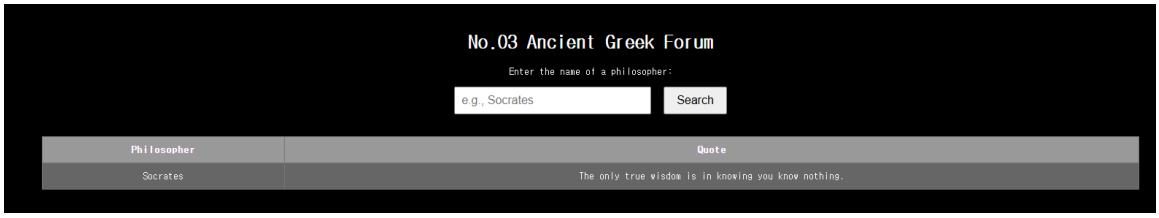
[그림 3-3] OWASP ZAP 을 통해 발견된 SQL 인젝션 취약점 1



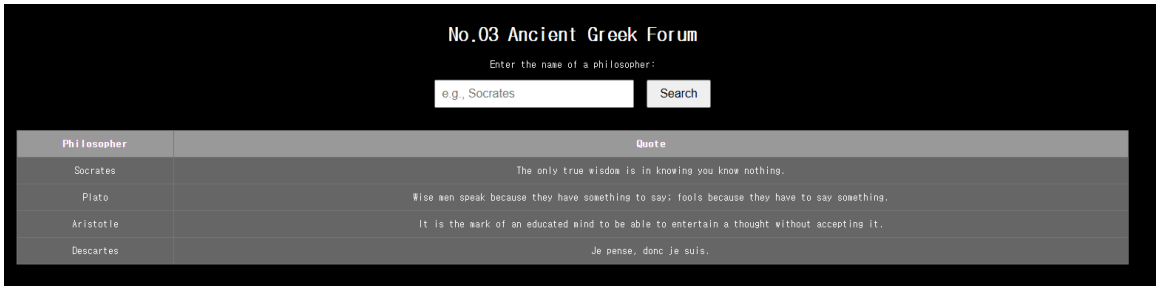
[그림 3-4] OWASP ZAP 을 통해 발견된 SQL 인젝션 취약점 2

웹의 name 입력값에 대해 SQL 인젝션 테스트를 수행한 결과, 사용자 입력이 서버 측 SQL 쿼리에 그대로 반영되어 필터링 없이 실행되는 취약점이 확인되었습니다.

(2) 수동 점검

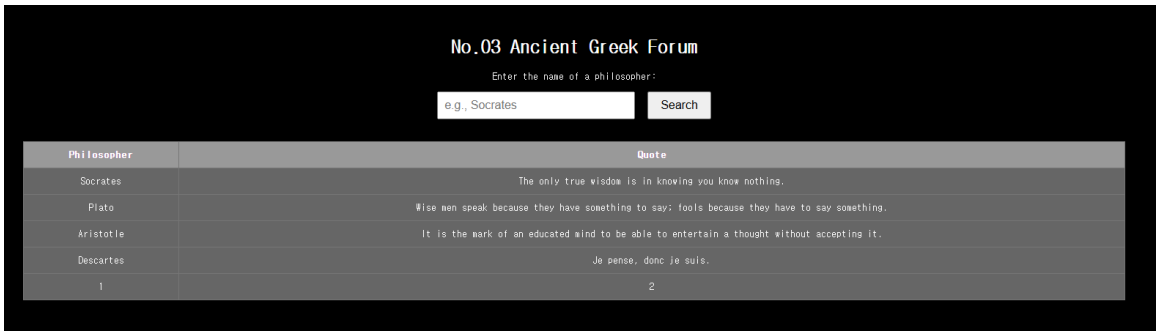


[그림 3-5] 정상 입력 결과



[그림 3-6] SQL 참 쿼리로 조작된 입력 결과

OWASP ZAP 을 통해 알아낸 SQL 인젝션 취약점을 바탕으로 항상 참이 되는 SQL 조건('OR 1=1#)을 삽입하였고, 서버는 이를 정상적으로 처리하여 결과를 반환하였습니다. 그 결과, 데이터베이스 내 모든 값이 출력되었습니다.

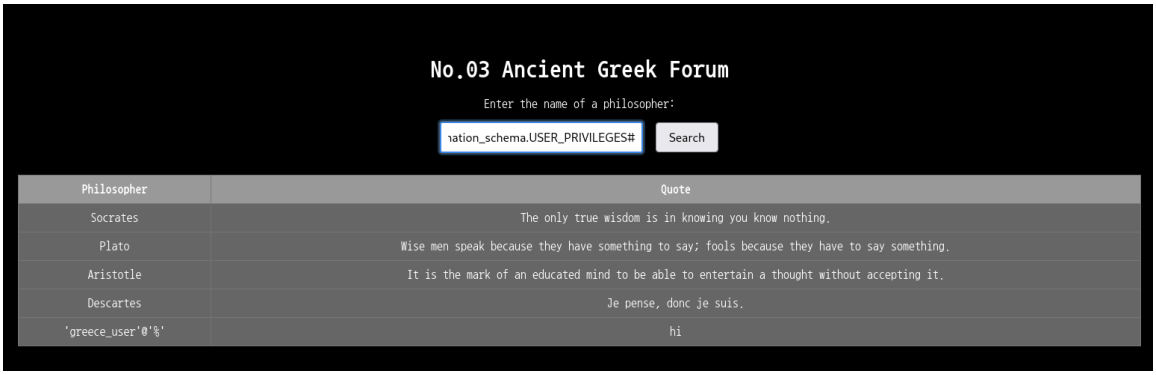


[그림 3-7] 열 개수 확인

데이터베이스 정보를 확인하기 위해 union select 문을 이용한 SQL 인젝션을 시도하였습니다. 이때 ' union select all 1,2 #를 입력하여, 해당 쿼리가 정상적으로 실행되는 것을 통해 데이터베이스 테이블의 출력 열 개수가 2 개임을 확인할 수 있었습니다.

[그림 3-8] 테이블명 확인

출력 열의 개수가 2 개임을 확인한 후, ' union select all table_name, 2 from information_schema.tables # 구문을 이용하여 데이터베이스 내에 philosophers 라는 테이블이 존재함을 확인하였습니다.



[그림 3-9] 데이터베이스 계정명 확인

' union select grantee, 'hi' from information_schema.user_privileges # 구문을 입력하여 데이터베이스 계정명(greece_user)을 확인할 수 있었습니다. 이 요청에 대한 응답이 화면에 그대로 출력되는 것을 통해, 내부 데이터베이스 테이블의 정보가 사용자에게 노출되는 취약한 구조임을 확인할 수 있었습니다.

해당 취약점은 권한 정보를 포함한 중요한 시스템 정보가 노출될 수 있는 위험이 있으며, 공격자는 이를 기반으로 추가적인 공격(예: 권한 상승, 인증 우회 등)을 시도할 수 있습니다.

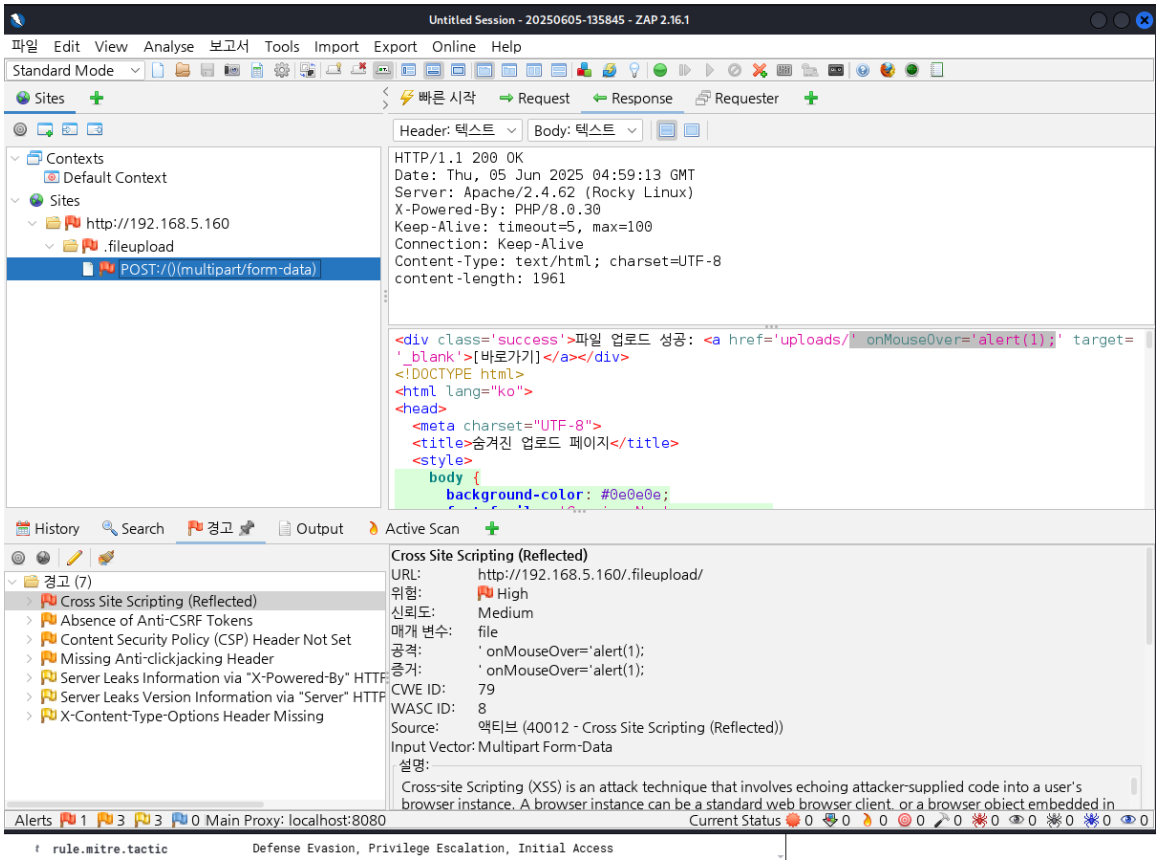
3.3 크로스사이트 스크립팅(XSS)

3.3.1 취약점 개념 설명

크로스사이트 스크립팅(XSS, Cross-Site Scripting)은 공격자가 웹사이트의 URL 이나 입력 폼 등에 악의적인 스크립트 코드를 삽입하고, 이를 통해 다른 사용자의 브라우저에서 해당 스크립트가 실행되도록 유도하는 공격 방식입니다. 이러한 공격은 사용자가 의도치 않게 공격자가 삽입한 코드를 실행하게 되어, 쿠키 탈취, 세션 하이재킹, 악성 행위 자동 실행 등의 피해로 이어질 수 있습니다.

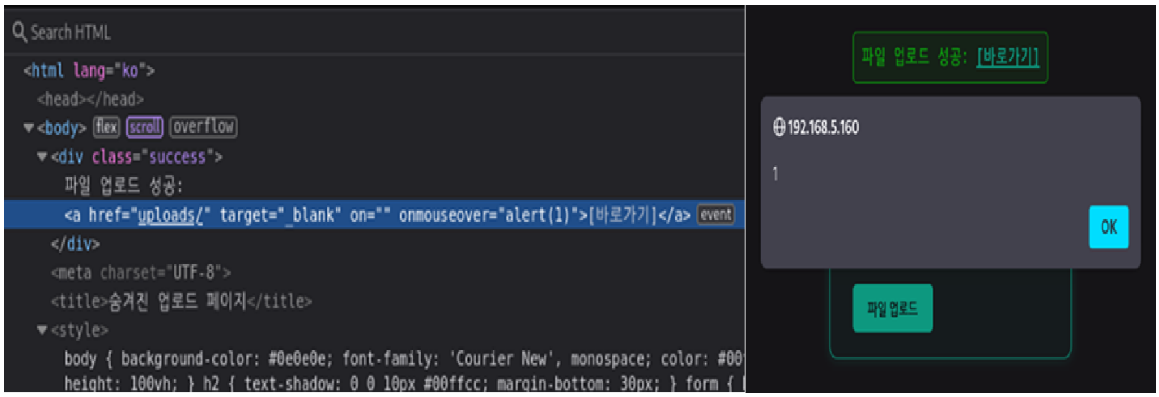
XSS 는 공격 방식에 따라 서버에 악성 스크립트가 저장되어 여러 사용자에게 영향을 미치는 저장형(Stored)과, 악성 스크립트가 요청 시점에 즉시 반영되어 동작하는 반사형(Reflected) 방식으로 나뉘며, 경우에 따라 DOM 을 조작하는 DOM 기반 XSS 도 존재합니다.

3.3.2 취약점 점검



[그림 3-10] OWASP ZAP 을 통해 발견된 XSS 취약점

OWASP ZAP 을 이용한 테스트 결과, /.fileupload 페이지에서 반사형 크로스사이트 스크립팅(이하 XSS) 취약점이 발견되었습니다.

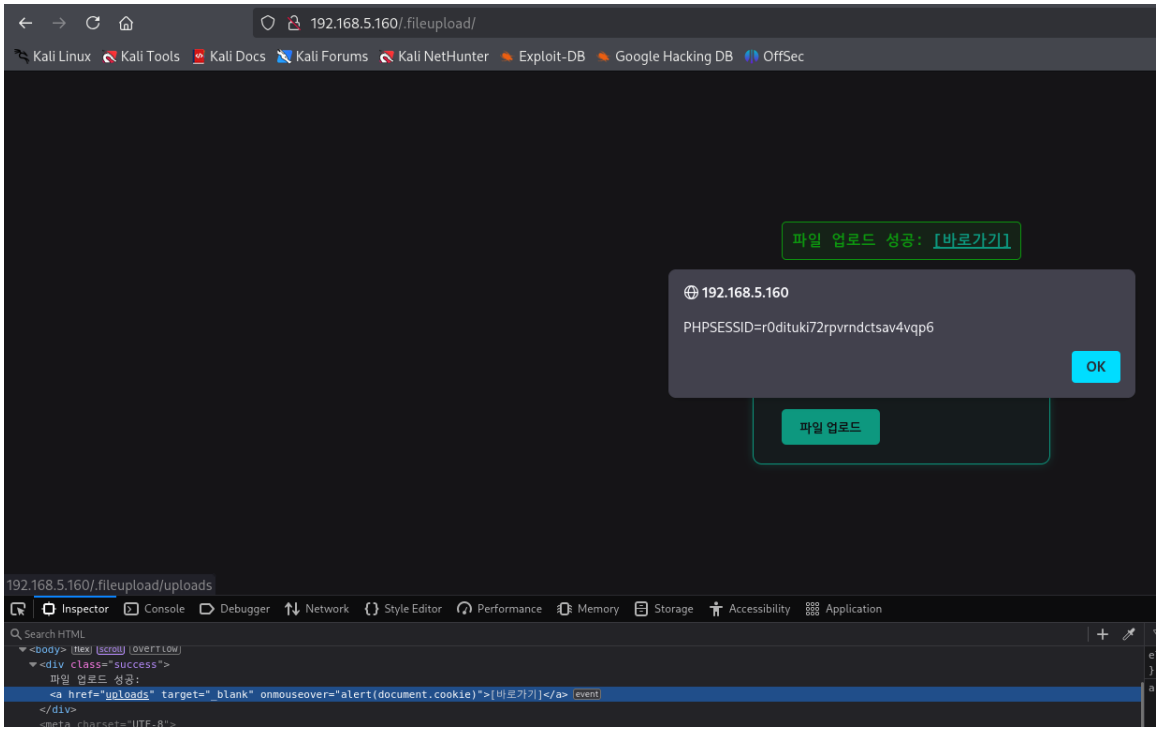


[그림 3-11] XSS 공격 테스트와 결과

실제 테스트를 통해 onMouseOver="alert(1)" 스크립트를 삽입한 결과, 업로드 성공 메시지 내 [바로가기] 링크에 마우스를 올리는 것만으로 악성 스크립트가 실행되는 현상이 확인되었습니다.

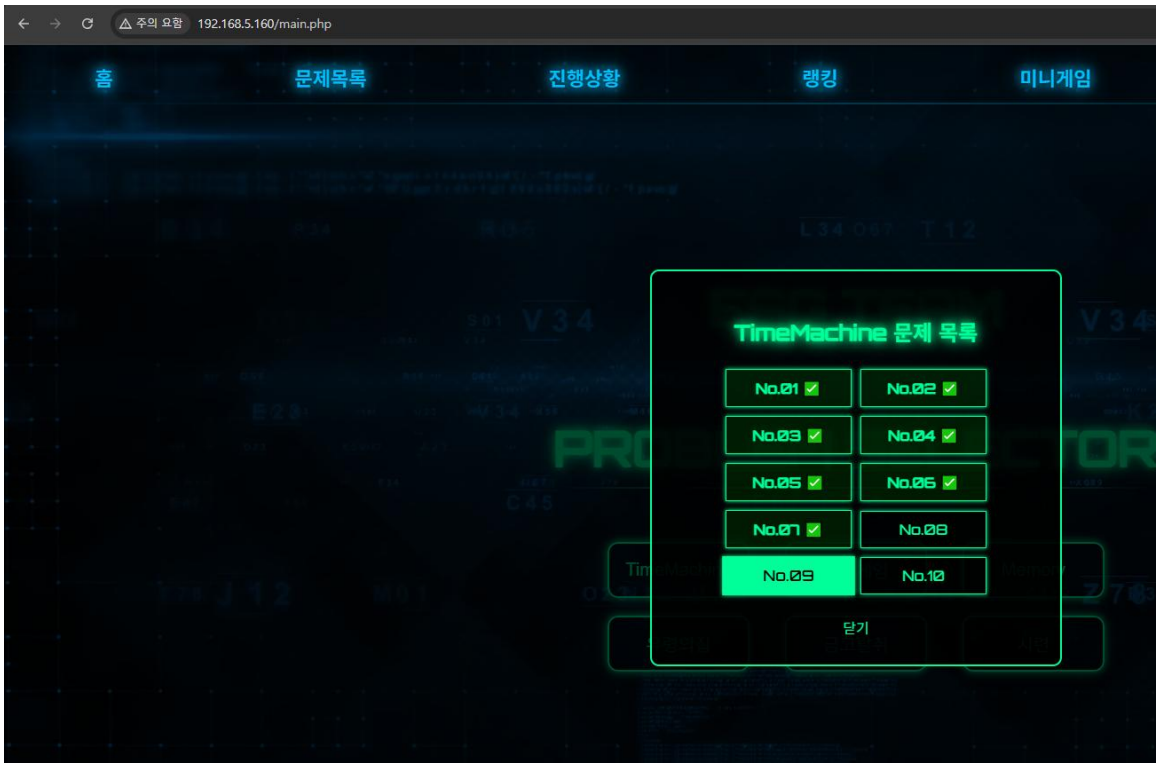
[그림 3-12] 쿠키값 변경 전

XSS 로 획득한 쿠키 값을 통해 세션 변조가 가능한지 확인하기 위해, 기존에 어떤 문제도 해결하지 않은 신규 계정을 생성하여 테스트를 진행하였습니다.



[그림 3-13] XSS 공격을 통한 쿠키값 탈취

일부 문제를 해결한 다른 계정으로 /.fileupload 페이지에 접속하여 onMouseOver="alert(document.cookie)" 스크립트를 삽입한 결과, 해당 계정의 쿠키 값을 획득할 수 있었습니다.



[그림 3-14] 쿠키값 변경 후

신규 계정으로 다시 로그인한 후, 획득한 쿠키 값으로 세션을 변경하고 문제 목록 페이지에 접근한 결과, 일부 문제를 해결한 계정의 권한으로 변경된 것이 확인되었습니다.

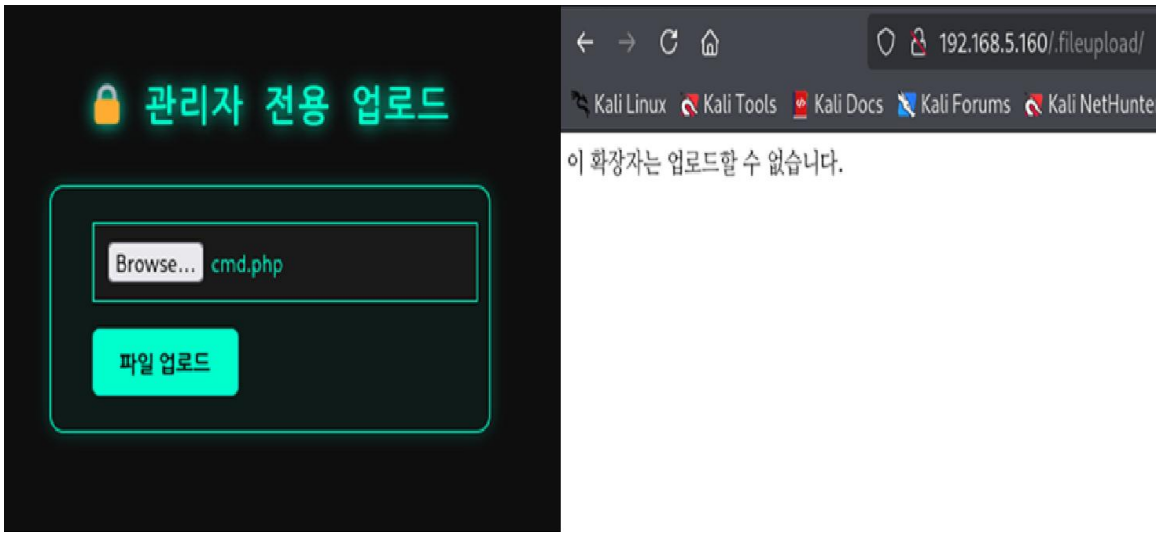
이를 통해 세션 탈취가 가능함을 알 수 있으며, 공격자는 동일한 방식으로 관리자 계정의 세션을 탈취해 일부 계정 정보를 변경하거나, 시스템을 장악하는 등의 추가 공격을 수행할 수 있습니다.

3.4 파일 업로드

3.4.1 취약점 개념 설명

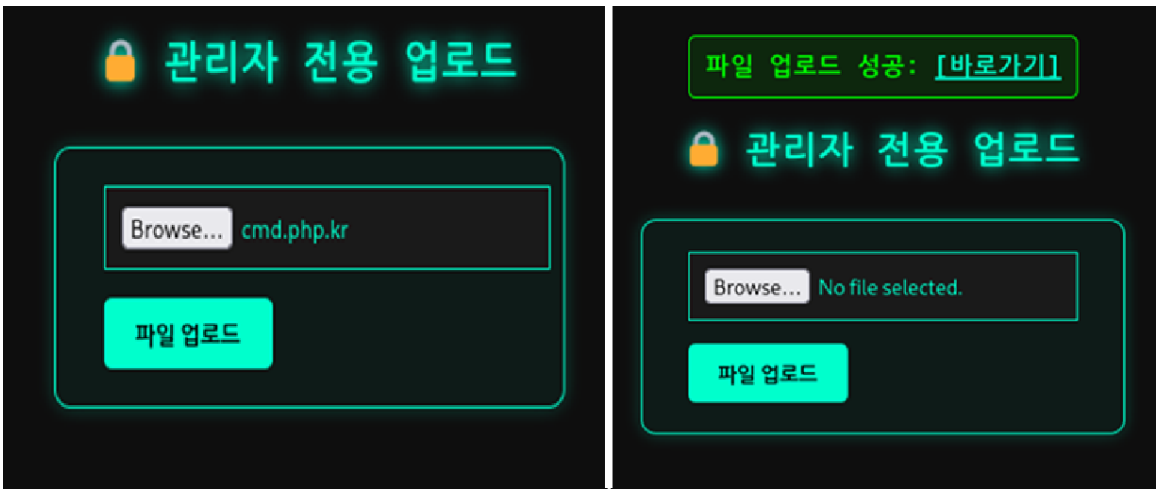
파일 업로드 취약점은 웹 게시판이나 자료실 등에서 파일을 첨부할 수 있는 기능을 악용하여 발생합니다. 일반적으로 허용된 파일 형식(txt, jpeg 등)이 아닌, 악의적인 서버 사이드 스크립트(jsp, php, asp 등)를 포함한 파일을 업로드한 후 이를 실행시켜 웹 서버에 쉘을 획득하거나 추가적인 악성 행위를 수행할 수 있는 보안 취약점을 의미합니다. 이 취약점은 업로드 기능이 있는 웹 페이지에서 조작된 스크립트 파일이 필터링 없이 저장되고 실행 가능한지를 점검함으로써 확인할 수 있습니다.

3.4.2 취약점 점검



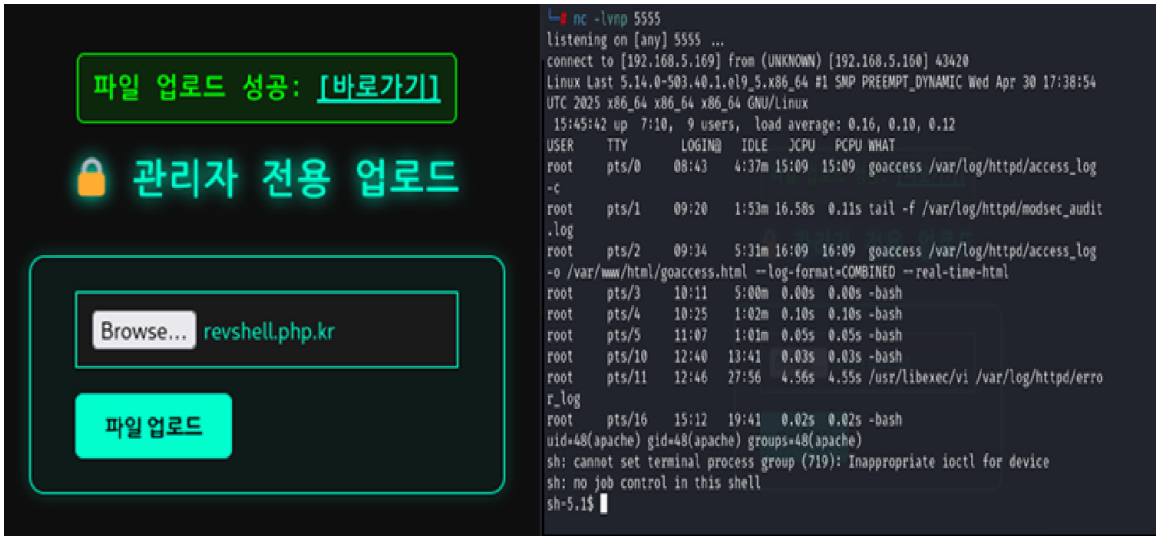
[그림 3-15] php 파일 업로드 시도와 결과

/.fileupload 페이지에서 cmd.php 파일을 업로드 시도한 결과, [그림 3-15]와 같이 "이 확장자는 업로드할 수 없습니다"라는 메시지가 출력되며 업로드가 차단되었습니다. 이는 서버 측에서 .php 와 같은 실행 가능한 스크립트 파일에 대한 확장자 필터링이 적용되어 있음을 보여줍니다.

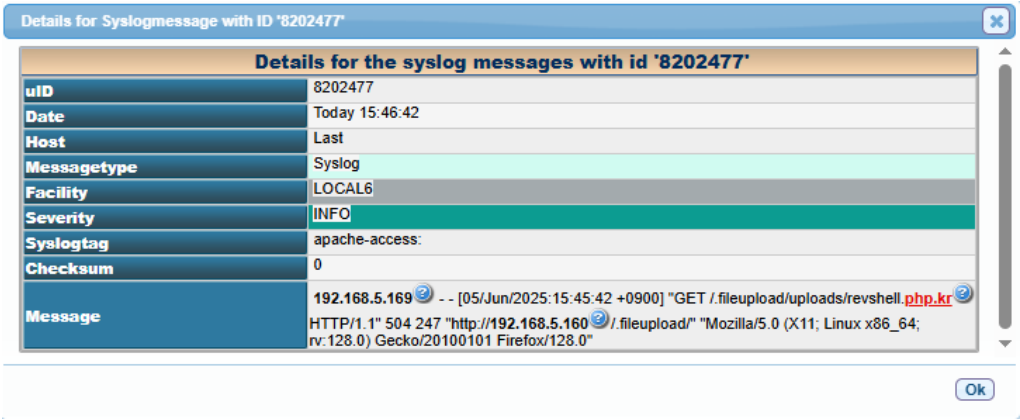


[그림 3-16] 확장자 우회 공격 시도와 결과

.php 가 차단되어 .php.kr 로 확장자를 우회해서 파일 업로드를 시도한 결과 업로드가 성공하였습니다.

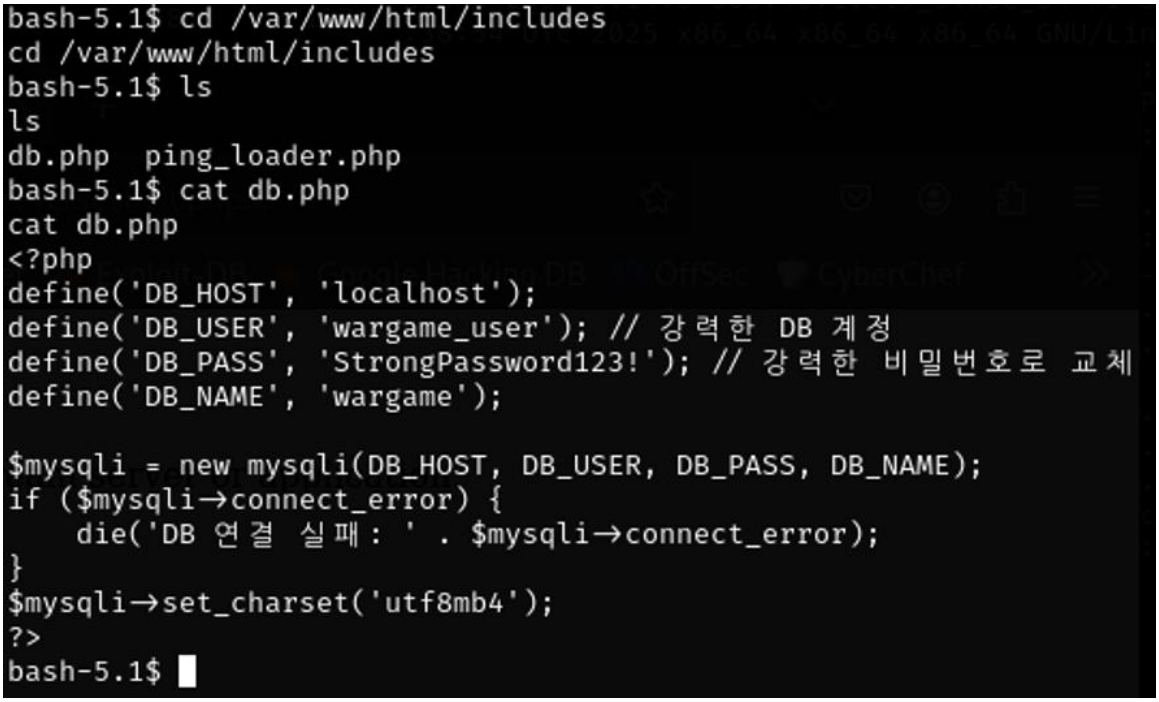


[그림 3-17] 리버스 쉘 시도와 결과



[그림 3-18] 리버스 쉘 파일 실행 로그 탐지

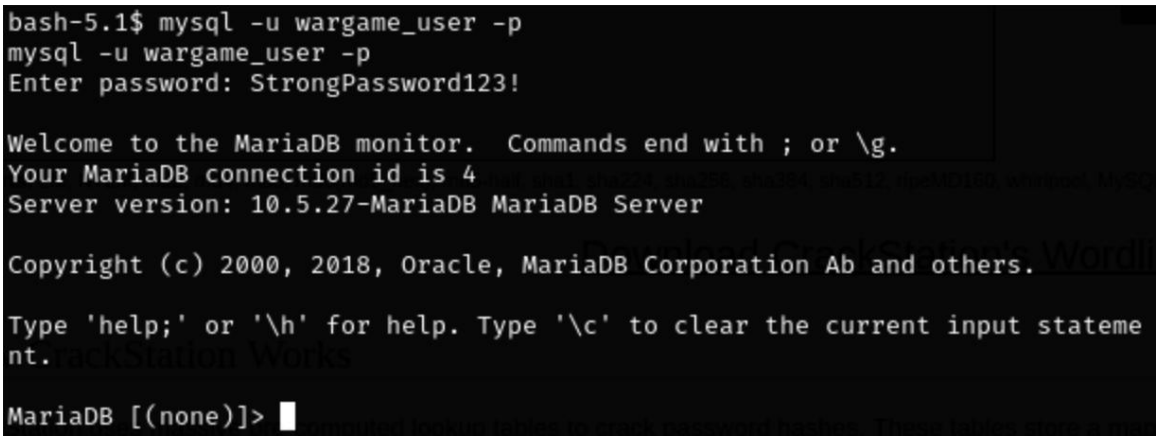
revshell.php.kr 파일에 리버스 쉘 코드를 작성한 뒤 업로드하여, Kali Linux 에서 nc -lvnp 5555 명령어로 리스닝 상태를 유지했습니다. 이후 해당 리버스 쉘 파일이 실행되면서, [그림 3-17]과 같이 192.168.5.160 서버에서 Kali Linux (192.168.5.169)로 연결이 시도되었고, 웹 서버의 Apache 권한으로 원격 쉘 획득에 성공한 모습을 확인할 수 있습니다.



[그림 3-19] 리버스 쉘을 통해 확인한 데이터베이스 접속 정보 파일

리버스 쉘을 통해 서버에 원격 접속한 후 내부 디렉토리를 탐색하던 중, 데이터베이스 계정과 비밀번호가 평문으로 저장된 db.php 파일을 확인하였습니다. 해당 정보는 인증 없이 데이터베이스에 직접 접근할 수 있는 수단이 되며, 계정 권한에 따라 데이터 조회, 수정, 삭제 등 심각한 추가 공격으로 이어질 수 있는 보안 위험 요소입니다.

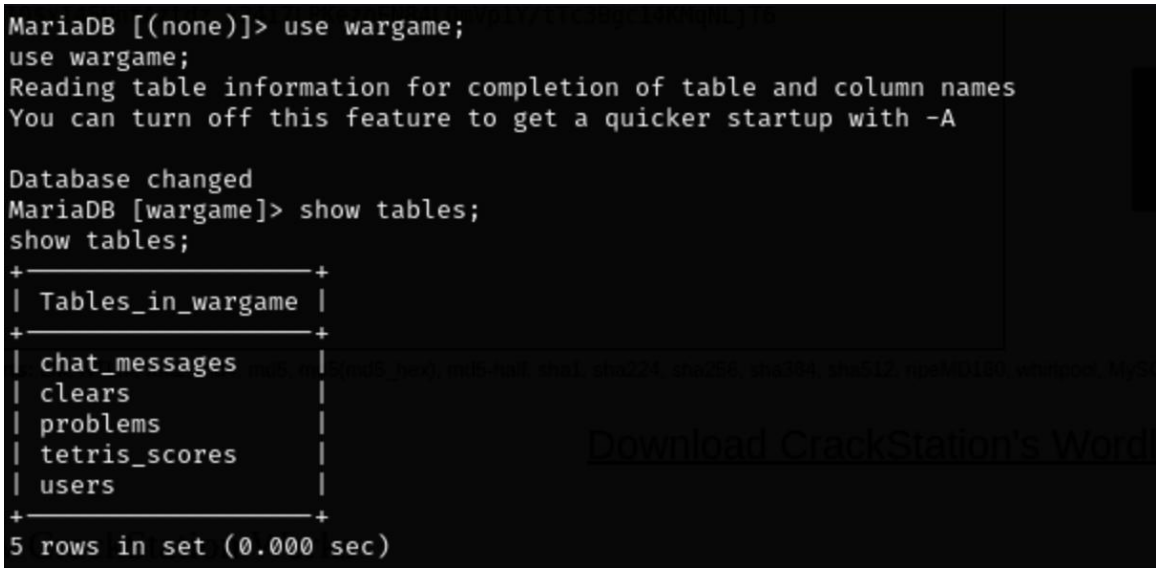
해당 정보는 인증 없이 데이터베이스에 직접 접근할 수 있는 수단이 되며, 계정 권한에 따라 데이터 조회, 수정, 삭제 등 심각한 추가 공격으로 이어질 수 있는 보안 위험 요소입니다.



[그림 3-20] 내부 데이터베이스 접근

db.php 파일에 저장된 데이터베이스 접속 정보를 활용하여 MariaDB 서버에 접근하는 데 성공하였습니다. 해당 파일을 통해 MySQL 클라이언트에서 인증을 우회하고 내부 데이터베이스에 직접 접근할 수 있었습니다.

접속 결과, MariaDB 서버의 버전은 10.5.27 이며, 사용자 인증 이후 MariaDB 쉘에 진입한 것을 확인할 수 있습니다.

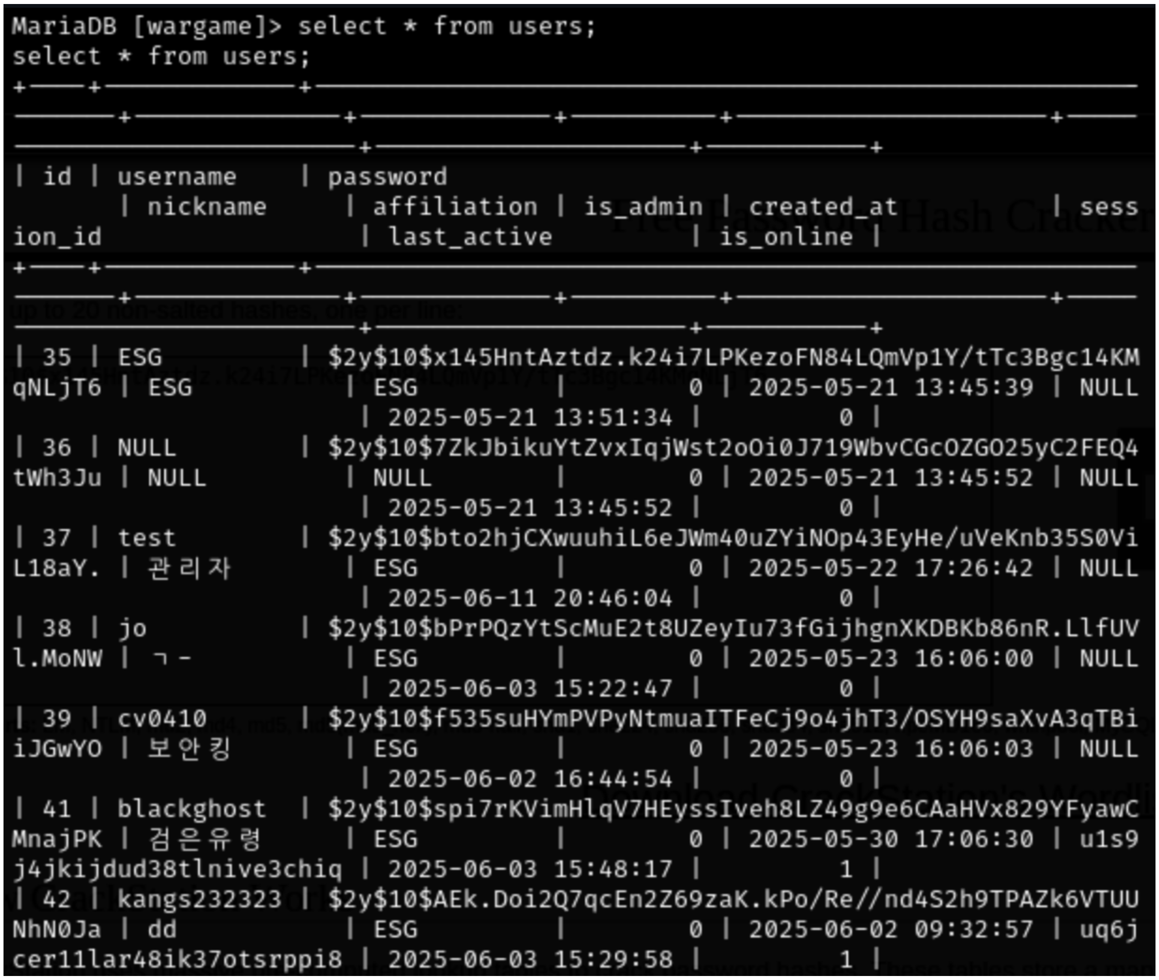


[그림 3-21] wargame 데이터베이스의 테이블 확인

wargame 데이터베이스에 접속한 후, 내부 테이블 목록을 확인한 결과 사용자 정보와 관련된 테이블들을 확인할 수 있었습니다.

use wargame; 명령어로 데이터베이스를 선택한 뒤, show tables; 명령어를 통해 chat_messages, clears, problems, tetris_scores, users 등 총 5 개의 테이블이 존재함을 확인하였습니다.

이 중 users 테이블은 워게임 사용자 계정 정보가 저장되어 있을 가능성이 높으며, 추가적인 정보 조회를 통해 계정명, 패스워드, 권한 등의 민감한 정보를 확인하거나 조작할 수 있는 위험이 존재합니다.



```
MariaDB [wargame]> select * from users;
select * from users;
+----+-------------+---------------------------------------------------------
-------+--------------+--------------+----------+-----------------------+------
--------------------+-----------------------+------------+
| id | username    | password
       | nickname     | affiliation | is_admin | created_at            | sess
ion_id                | last_active           | is_online |
+----+-------------+---------------------------------------------------------
-------+--------------+--------------+----------+-----------------------+------
--------------------+-----------------------+------------+
| 35 | ESG         | $2y$10$x145HntAztdz.k24i7LPKezoFN84LQmVp1Y/tTc3Bgc14KM
qNLjT6 | ESG          | ESG          |        0 | 2025-05-21 13:45:39 | NULL
                      | 2025-05-21 13:51:34 |         0 |
| 36 | NULL        | $2y$10$7ZkJbikuYtZvxIqjWst2oOi0J719WbvCGcOZGO25yC2FEQ4
tWh3Ju | NULL         | NULL         |        0 | 2025-05-21 13:45:52 | NULL
                      | 2025-05-21 13:45:52 |         0 |
| 37 | test        | $2y$10$bto2hjCXwuuhiL6eJWm40uZYiNOp43EyHe/uVeKnb35S0Vi
L18aY. | 관리자       | ESG          |        0 | 2025-05-22 17:26:42 | NULL
                      | 2025-06-11 20:46:04 |         0 |
| 38 | jo          | $2y$10$bPrPQzYtScMuE2t8UZeyIu73fGijhgnXKDBKb86nR.LlfUV
l.MoNW | ㄱ-           | ESG          |        0 | 2025-05-23 16:06:00 | NULL
                      | 2025-06-03 15:22:47 |         0 |
| 39 | cv0410      | $2y$10$f535suHYmPVPyNtmuaITFeCj9o4jhT3/OSYH9saXvA3qTBi
iJGwYO | 보안킹       | ESG          |        0 | 2025-05-23 16:06:03 | NULL
                      | 2025-06-02 16:44:54 |         0 |
| 41 | blackghost  | $2y$10$spi7rKVimHlqV7HEyssIveh8LZ49g9e6CAaHVx829YFyawC
MnajPK | 검은유령     | ESG          |        0 | 2025-05-30 17:06:30 | u1s9
j4jkijdud38tlnive3chiq | 2025-06-03 15:48:17 |         1 |
| 42 | kangs232323 | $2y$10$AEk.Doi2Q7qcEn2Z69zaK.kPo/Re//nd4S2h9TPAZk6VTUU
NhN0Ja | dd           | ESG          |        0 | 2025-06-02 09:32:57 | uq6j
cer11lar48ik37otsrppi8 | 2025-06-03 15:29:58 |         1 |
```

[그림 3-22] users 테이블 조회

users 테이블의 전체 데이터를 조회한 결과, 워게임 사용자들의 계정 정보가 포함된 민감한 데이터들이 확인되었습니다.

각 사용자에 대해 username, password, affiliation, is_admin, created_at, last_active, is_online, session_id 등의 필드가 존재하며, 이 중 password 항목에는 bcrypt 해시 형태로 암호화된 비밀번호가 저장되어 있습니다.

is_admin 필드를 통해 관리자 여부를 식별할 수 있으며, 세션 ID(session_id)와 마지막 접속 시간(last_active) 등의 정보도 함께 포함되어 있어, 세션 하이재킹 등 추가적인 공격 가능성이 존재합니다.

이러한 정보 노출은 중대한 보안 사고로 이어질 수 있으며, 비밀번호 크랙을 시도하거나 직접적인 세션 탈취를 통한 인증 우회 공격이 가능해질 수 있습니다.

[그림 3-23] 비밀번호 크랙

사용자 계정의 비밀번호가 충분히 복잡하지 않을 경우, [그림 3-23]과 같이 손쉽게 크랙될 수 있습니다.

해당 이미지에서는 john 툴로 크랙을 시도하였으며, 비밀번호가 1234 임을 확인할 수 있었습니다.

따라서 비밀번호 복잡도 정책 적용과 주기적인 비밀번호 변경, 계정 잠금 정책 등이 반드시 필요합니다. 또한, 데이터베이스 관련 정보는 반드시 안전한 방식으로 보호되어야 하며, 비밀번호는 암호화하여 저장하고, 파일 자체는 접근 제어를 통해 외부에 노출되지 않도록 해야 합니다.

4. 대응방안

4.1 XSS 대응 방안

① 입력 값에 대한 사전 필터링 및 태그 제한

사용자 입력을 저장하거나 출력하는 웹 페이지에서는 <script>, <iframe>, <form>, <object> 등 실행 가능한 스크립트 관련 태그와 HTML 태그의 사용을 제한하고, 서버 측에서 입력 값을 철저히 검증해야 합니다.

② 모든 사용자 입력 영역에 대한 필터링 적용

게시물 본문뿐 아니라 제목, 댓글, 검색어 입력창, 회원정보 변경, 자료 업로드, URL 파라미터 등 사용자가 입력하거나 전송할 수 있는 모든 form 요소와 파라미터에 대해 필터링을 적용해야 합니다.

③ 서버 측 필터링 로직 구현

입력 값 처리 시 Java, PHP, Python 등 백엔드 언어에서 trim(), replace() 등의 함수를 사용하여 공백 문자, 특수문자, 위험한 키워드를 제거하는 필터링 로직을 서버 측에 구현해야 합니다. 클라이언트 측(JavaScript)만으로는 충분하지 않습니다.

④ URL 인코딩 우회 차단

URLDecoder.decode()와 같은 메서드를 이용해 인코딩된 사용자 입력 값을 디코딩한 뒤 필터링함으로써, %3Cscript%3E 와 같은 우회 공격도 차단할 수 있습니다.

⑤ 웹 방화벽(WAF)을 통한 특수문자 탐지 및 차단

웹 방화벽에 회원가입, 게시판, 댓글, 검색 등 모든 입력 항목에 대해 <, >, ", ', &, %, %00(null) 등의 특수문자나 스크립트 키워드가 포함된 요청을 탐지하고 차단하는 룰셋을 구성해야 합니다.

4.2 SQL Injection 대응 방안

① 입력 값에 대한 유효성 검증 및 필터링 적용

사용자 입력이 SQL 쿼리에 포함되기 전에, 입력 값에 대한 유효성 검증을 수행해야 합니다. 숫자, 이메일, ID 등 특정 형식을 요구하는 값에 대해서는 사전에 허용된 패턴만 통과하도록 서버 측 검증 로직을 구현해야 합니다.

② 위험 특수문자 필터링

SQL 문법에서 사용되는 특수문자를 사용자 입력으로부터 차단하거나 적절히 이스케이프 처리해야 합니다. ', ;, -- , #, /* */ 등 특수문자는 데이터베이스 종류(MySQL, MSSQL, Oracle 등)에 따라 다르게 작동할 수 있으므로, 환경에 맞는 필터링 정책이 필요합니다.

③ 예외 처리 및 에러 메시지 노출 방지

쿼리 실행 중 발생하는 예외에 대해 적절한 예외처리를 구현하여, 시스템 내부 에러 메시지나 DBMS 에서 제공하는 에러 코드, 쿼리 구조 등이 사용자에게 노출되지 않도록 해야 합니다. 에러 정보를 그대로 노출할 경우, 공격자에게 시스템 구조를 유추할 수 있는 단서를 제공하게 됩니다.

4.3 파일 업로드 취약점 대응 방안

사용자 파일 업로드가 가능한 모든 모듈에 대해 동일한 보안 정책이 적용되도록 관리하며, 취약점이 발생하지 않도록 철저한 점검이 필요합니다.

① 허용된 확장자만 업로드 가능한 화이트리스트 방식 적용

서버 측에서는 사전에 정의된 안전한 확장자만 업로드할 수 있도록 화이트리스트 기반의 확장자 필터링을 적용해야 하며, .php, .jsp, .exe 등 실행 가능한 확장자는 업로드 대상에서 철저히 제외합니다.

② 파일명 및 확장자 무작위화 처리

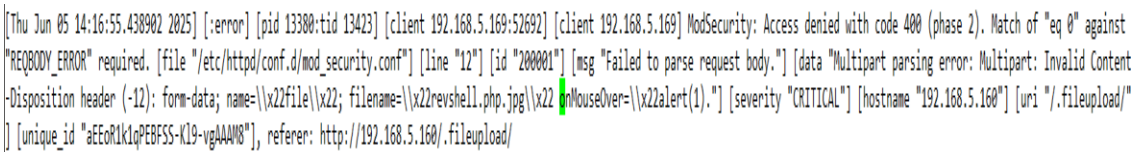
업로드된 파일은 외부 사용자가 직접 접근하거나 추측할 수 없도록 무작위 문자열로 파일명을 변경하여 저장하고, 실제 파일명은 데이터베이스에 별도로 저장 및 관리합니다.

③ 업로드 전용 디렉터리 분리 및 실행 권한 제거

업로드된 파일을 보관할 전용 디렉터리를 별도로 구성하고, 해당 디렉터리에 대해 httpd.conf, .htaccess 등 웹 서버 설정 파일을 통해 스크립트 실행이 불가능하도록 설정함으로써, 서버 사이드 스크립트가 업로드되더라도 실행되지 않도록 차단합니다.

④ 웹 방화벽(WAF)을 통한 특수문자 필터링

업로드 필드를 대상으로 웹 방화벽(WAF) 룰을 설정하여 <, >, ;, ?, % 등 특수문자를 포함한 악성 입력을 탐지 및 차단함으로써 우회 시도를 방지합니다.



[그림 4-1] WAF 룰에 의해 탐지된 파일 업로드에 대한 로그